MLP Classification (Iris dataset)

1. Jumlah Hidden Layer

Jumlah hidden layer sangat berpengaruh terhadap kemampuan model untuk mempelajari pola yang kompleks.

- **1 Hidden Layer**: Biasanya cukup untuk dataset sederhana seperti Iris, karena masalah ini adalah linier atau hampir linier separable. Namun, terlalu sedikit hidden layer dapat menyebabkan model kurang mampu menangkap pola kompleks.
- **2 atau 3 Hidden Layers**: Menambah hidden layer memungkinkan model menangkap pola yang lebih abstrak, tetapi pada dataset kecil seperti Iris, ini mungkin hanya sedikit meningkatkan performa atau bahkan menurunkan jika terjadi overfitting.

Sebagai rekomendasi, untuk dataset kecil seperti Iris, jumlah hidden layer yang lebih sederhana (1–2 layer) sudah mencukupi.

Berikut adalah analisis untuk setiap hyperparameter yang digunakan dalam eksperimen menggunakan **MLP (Vanilla Multi-Layer Perceptron)**:

---

1. Jumlah Hidden Layer

Jumlah hidden layer sangat berpengaruh terhadap kemampuan model untuk mempelajari pola yang kompleks.

- **1 Hidden Layer**: Biasanya cukup untuk dataset sederhana seperti Iris, karena masalah ini adalah linier atau hampir linier separable. Namun, terlalu sedikit hidden layer dapat menyebabkan model kurang mampu menangkap pola kompleks.
- **2 atau 3 Hidden Layers**: Menambah hidden layer memungkinkan model menangkap pola yang lebih abstrak, tetapi pada dataset kecil seperti Iris, ini mungkin hanya sedikit meningkatkan performa atau bahkan menurunkan jika terjadi overfitting.

Sebagai rekomendasi, untuk dataset kecil seperti Iris, jumlah hidden layer yang lebih sederhana (1–2 layer) sudah mencukupi.

2. Jumlah Neuron pada Hidden Layer

Jumlah neuron menentukan kapasitas representasi setiap layer.

- **Jumlah Neuron Kecil (4, 8)**: Jumlah kecil mungkin tidak cukup untuk menangkap pola yang relevan, terutama jika data memiliki hubungan yang lebih kompleks.

- **Jumlah Neuron Besar (32, 64, dst.)**: Menambah jumlah neuron meningkatkan kapasitas model, tetapi bisa menyebabkan overfitting jika dataset terlalu kecil. Selain itu, peningkatan jumlah neuron akan meningkatkan waktu komputasi.

Pada dataset kecil seperti Iris, jumlah neuron yang moderat (8–32) biasanya memberikan hasil yang optimal tanpa terlalu membebani model.

3. Fungsi Aktivasi

Fungsi aktivasi bertugas menambahkan non-linearitas ke dalam model, yang penting untuk menangkap hubungan non-linear dalam data.

- **Linear**: Fungsi linear jarang digunakan karena tidak menambahkan non-linearitas, sehingga kurang efektif dalam mempelajari hubungan kompleks.
- **Sigmoid**: Cocok untuk output probabilitas, tetapi memiliki kelemahan vanishing gradient pada jaringan yang dalam.
- **ReLU**: Fungsi aktivasi yang populer, sederhana, dan efektif. Cocok untuk sebagian besar kasus karena mempercepat konvergensi dan mengurangi masalah vanishing gradient.
- **Tanh**: Alternatif Sigmoid dengan output dalam rentang [−1,1][-1, 1][−1,1], dapat membantu dalam beberapa kasus, tetapi juga rentan terhadap vanishing gradient.
- **Softmax**: Biasanya diterapkan di layer output untuk klasifikasi multi-kelas.

Pada eksperimen ini, **ReLU** sering menjadi pilihan terbaik untuk hidden layer karena efisiensinya dalam melatih model.

4. Jumlah Epoch

Epoch menentukan seberapa banyak model melihat keseluruhan dataset selama proses pelatihan.

- **Epoch Kecil (1, 10)**: Tidak cukup untuk melatih model sepenuhnya, terutama pada dataset yang membutuhkan iterasi lebih banyak untuk mencapai konvergensi.
- **Epoch Sedang (50–100)**: Memberikan keseimbangan antara waktu pelatihan dan performa.
- **Epoch Besar (250)**: Memberikan model lebih banyak waktu untuk belajar, tetapi dapat menyebabkan overfitting jika model terus belajar dari noise dalam data.

Sebagai rekomendasi, menggunakan jumlah epoch sekitar 50–100 sering kali cukup untuk dataset kecil seperti Iris.

5. Learning Rate

Learning rate menentukan ukuran langkah dalam proses pembaruan bobot selama pelatihan.

- **Learning Rate Besar (10, 1)**: Mempercepat proses pelatihan, tetapi risiko besar model melewati solusi optimal.
- **Learning Rate Moderat (0.1, 0.01)**: Umumnya paling efektif untuk mencapai keseimbangan antara kecepatan pelatihan dan akurasi.
- **Learning Rate Kecil (0.001, 0.0001)**: Memberikan pembaruan yang lebih hati-hati, tetapi pelatihan bisa menjadi sangat lambat.

Learning rate **0.01 atau 0.001** biasanya optimal karena memberikan pembaruan yang cukup stabil untuk konvergensi.

6. Batch Size

Batch size menentukan jumlah sampel yang digunakan untuk memperbarui bobot setiap iterasi.

- **Batch Kecil (16, 32)**: Mempercepat pembaruan model, tetapi sering menyebabkan fluktuasi tinggi pada proses pelatihan.
- **Batch Sedang (64, 128)**: Memberikan keseimbangan antara kecepatan pelatihan dan stabilitas.
- **Batch Besar (256, 512)**: Meningkatkan stabilitas pembaruan tetapi membutuhkan memori lebih besar, yang bisa menjadi masalah pada dataset besar.

Untuk dataset seperti Iris, batch size sedang (64–128) sering kali memberikan performa terbaik, karena ukuran dataset kecil dan tidak memerlukan komputasi intensif.